UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
09/874,170	06/04/2001	Vasanth Bala	10003355-1	7644

7590 08/28/2006

HEWLETT-PACKARD COMPANY
Intellectual Property Administration
P.O. Box 272400
Fort Collins, CO 80527-2400

EXAMINER	
PROCTOR, JASON SCOTT	

ART UNIT	PAPER NUMBER
2123	

DATE MAILED: 08/28/2006

Please find below and/or attached an Office communication concerning this application or proceeding.

Office Action Summary	Application No.	Applicant(s)	
	09/874,170	BALA ET AL.	
	Examiner	Art Unit	
	Jason Proctor	2123	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *05 June 2006*.

2a)☒ This action is **FINAL**. 2b)☐ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *2-24* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *2-24* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☒ The drawing(s) filed on *04 June 2001* is/are: a)☒ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☐ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☐ Information Disclosure Statement(s) (PTO-1449 or PTO/SB/08) Paper No(s)/Mail Date _____.
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____.
5) ☐ Notice of Informal Patent Application (PTO-152)
6) ☐ Other: _____.

DETAILED ACTION

Claims 1-24 were rejected in office action dated 9 May 2006. Applicants' response dated 5 June 2006 has amended claims 2, 6, 9, 14, 18, and 23; and cancelled claim 1. Claims 2-24 are pending in the application.

Claims 2-24 have been rejected.

Claim Interpretation

1. The language of claims 1, 9, 14, 18, and 23 (now claims 2, 9, 14, 18, and 23) have been previously objected for the claim language "wherein a granularity of the code segments is dynamically tailored to the client to balance server-side and client-side execution overhead and network bandwidth efficiency and client-side storage requirements". Applicants remarks submitted on 9 August 2006 refer to the specification regarding this limitation.

The Examiner reasserts that it is not entirely clear what is meant by "a granularity of the code segment," or whether every code segment has a "granularity". This terminology appears to be unconventional. It is not entirely clear how to identify whether the granularity of a particular code segment has been "dynamically tailored". It is unclear if this distinguishes the granularity from other types of tailoring or other dynamic processes.

The language "to balance server-side and client-side execution overhead and network bandwidth efficiency and client-side storage requirements" merely states an intention and does not limit the claim. Prior art that "dynamically tailors a granularity of the code segments" for some other purpose would anticipate this limitation.

Claim Rejections - 35 USC § 112

The previous rejection of claims 1-24 under 35 U.S.C. § 112, first paragraph, as failing to comply with the written description requirement, is withdrawn in response to Applicants' remarks.

Claim Rejections - 35 USC § 103

The following is a quotation of 35 U.S.C. § 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

The factual inquiries set forth in *Graham* **v.** *John Deere Co.*, 383 U.S. 1, 148 USPQ 459 (1966), that are applied for establishing a background for determining obviousness under 35 U.S.C. § 103(a) are summarized as follows:

1. Determining the scope and contents of the prior art.
2. Ascertaining the differences between the prior art and the claims at issue.
3. Resolving the level of ordinary skill in the pertinent art.
4. Considering objective evidence present in the application indicating obviousness or nonobviousness.

This application currently names joint inventors. In considering patentability of the claims under 35 U.S.C. § 103(a), the examiner presumes that the subject matter of the various claims was commonly owned at the time any inventions covered therein were made absent any evidence to the contrary. Applicant is advised of the obligation under 37 CFR 1.56 to point out the inventor and invention dates of each claim that was not commonly owned at the time a later

invention was made in order for the examiner to consider the applicability of 35 U.S.C. § 103(c) and potential 35 U.S.C. § 102(e), (f) or (g) prior art under 35 U.S.C. § 103(a).

1. Claims 2-20 are rejected under 35 U.S.C. § 103(a) as being unpatentable over US Patent No. 6,370,687 to Shimura (supplied by Applicant) and further in view of Official Notice.

Regarding claim 2, Shimura teaches:

A networked system (column 2, lines 16-30) comprising:

a network (column 2, lines 16-30):

a server coupled to the network, wherein the server includes (column 2, lines 16-30):

an application code source that stores a client application, and a server code manager coupled to the application code source (Web server from which programs are retrieved, column 4, lines 27-35; Fig. 1, reference 20);

an application code transformation manager coupled to the application code source, for transforming the client application from a first format to a native binary format compatible with a native instruction set of the CPU of the client ("compile controller ... recognizes the execute form of the client ... and recognizes the native code of the execution environment in which the compile unit 28 compiles from the byte code of the Java program"; column 4, line 52 – column 5, line 26); and

a server code segment manager coupled to the application code transformation manager, for parsing the client application in the native binary format into a plurality of code segments, wherein a granularity of the code segments is dynamically tailored to the client to balance server-side and client-side execution overhead and network bandwidth efficiency and client-side storage

requirements, and is configured based on predicted code segment usage or prior code segment usage history, wherein at least one of said plurality of code segments is transmitted to the client via the network [*"Then, in response to a request for the Java program from a client to the Web server on the network, the substitute compile server 10 returns to the client the Java program which has been* ***compiled and optimized into a native code conforming to the execute form of the requester client.***" (column 6, lines 25-48, emphasis added)]; and

a client coupled to the network said client not having said client application stored thereon, wherein the client comprises (column 2, lines 16-30):

a CPU for natively executing at least one of said plurality of said code segments derived from the client application stored on said server (column 4, line 52 – column 5, line 26);

a code cache coupled to the CPU for storing said code segments (Official notice is taken that it is well known in the art to provide a CPU with a code cache. This Official Notice was first taken in the office action of 16 May 2005. Applicants have not traversed this use of Official Notice and as such, it is regarded as admitted prior art. See MPEP 2144.03 (C)); and

a client code manager-coupled to the code cache, wherein the client code manager launches the client application by requesting that the server code manager transmit at least one of the plurality of code segments to the client (column 5, lines 27-32), receiving at least one of the code segment from the server (column 5, lines 58-61), storing the code segment in the code cache (Official Notice has been taken above regarding a code cache), and executing at least one of the plurality of code segments using the CPU until the executed code segment attempts to pass control to a required code segment not stored in the code cache (column 7, line 42 – column 8, line 18), at which point control passes back to the client code manager to retrieve the required

code segment from the server, with the CPU continuing execution with the required code segment [*"...if the client side makes a request for the class file 42-2* [a next required code segment] *with successful prediction, then the compiled Java program can be executed at a high speed in an immediate response to the program request from the client side sine that program has already been compiled and retained on the cache unit 12."* (column 7, line 42 – column 8, line 18)].

It would have been obvious to a person of ordinary skill in the art at the time of Applicants' invention to implement the system taught by Shimura using a code cache in conjunction with the list of processors expressly taught by Shimura [*"SPARCV8, X86, Intel 486, etc."* (column 4, line 52 – column 5, line 26) because these processors either possess a code cache or it is well known in the art to use a code cache. The motivation for using a code cache is well known in the art, for example, to increase the effective speed of the processor by caching instruction code on the processor.

In response, Applicants argue primarily that:

> Applicants respectfully submit that the claimed feature of the code being provided in native binary format to the CPU is not taught or rendered obvious by Shimura. That is, Applicants understand Shimura to teach virtual machine code operations.

The Examiner respectfully traverses this argument as follows.

Shimura teaches:

["The substitute compile server 10 using the Proxy server comprises ... a compile unit 28..." (column 4, lines 23-27; FIG. 2); "... the compile unit 28 as shown in FIG. 5 includes JIT compilers 38-1, 38-2, and 38-3 which have been prepared in advance and which correspond respectively to client execute forms α, β, and γ, so that it selects a JIT compiler in the execute

form corresponding to the requested server name **to thereby carry out compiling from a Java program byte code into a native code in the corresponding execute form**. For example, the JIT compiler 38-1 is a compiler for translating the Java byte code into a SPARCV8 native code, the JIT compiler 38-2 is a compiler for translating the Java byte code into an X86 native code, and the JIT compiler 38-3 is a compiler for translating the Java byte code into an Intel486 native code." (column 5, lines 7-26, emphasis added)]

Therefore the Examiner understands the compile server 10 to compile that Java applications into the native binary format (native code in the corresponding execute form) for each respective client and transferring that native binary format code to the client as claimed.

Applicants further argue that:

> Applicants do not understand Shimura to teach dynamic parsing of the application code into code segments. That is, assuming arguendo that Shimura parses application code into code segments. Applicants do not understand Shimura to teach parsing of code into code segments dynamically based on client and client server features.

The Examiner respectfully traverses this argument as follows.

It is not entirely clear what is meant by "dynamically parsing of the application code into code segments." Shimura plainly teaches dynamically compiling the application code, that is, compiling the code based on a dynamic request for the application (column 2, lines 10-29). A person of ordinary skill in the art of software compiling would recognize compiling application code as "parsing of the application code into code segments." Indeed the Examiner is unaware of any compiler that produces native binary code without "parsing of the application code into code segments." Further, Shimura teaches compiling of the application code based on client and

server features [client features (column 4, line 64 – column 5, line 26); server features such as cache (column 5, line 27 – column 6, line 48)].

Applicants further argue that:

> As the Examiner has stated on page 9 of the present Office Action, the Examiner cannot refute that Shimura does not teach storing the code segments on the client. The Examiner further states that the relevance is not understood.
>
> Applicants respectfully states that the relevance of the claimed features is clearly defined in the specification including the Summary section and the last three paragraphs of page 25. [...] As such, Applicants respectfully submit, as the Examiner has stated, Shimura does not teach or render obvious the claimed features and as such Claim 2 overcomes the rejection under 35 U.S.C. § 103(a).

The Examiner respectfully traverses this argument as follows.

The Examiner respectfully submits that Applicants have misconstrued the Examiner's statements. In the previous response, Applicants argued that:

> Shimura does not teach storing the class files on the client CPU. Instead, Applicant understands Shimura to teach storing the class files on the substitute compiler server thereby being available for a plurality of clients. (Response of 22 August 2005, page 14)

The Office Action stated:

> The Examiner cannot refute that Shimura does not teach storing the class files on the client, but does not understand the relevance of that observation. The class files of Shimura are compiled to produce code segments [*"[I]n response to a request ... from a client ... the substitute compile server 10 returns to the client the Java program which has been compiled and optimized into a native code conforming to the execute form of the requester client."* (column 6, lines 25-30)]. As shown above, those code segments are transferred to the client for execution. None of the claims appear to require "storing class files on the client." (Office Action of 2 March 2006, page 9)

The Examiner's statement was directed to Applicants' argument, the relevance of which is still not understood. The claims neither require nor exclude "storing class files on the client." As the Examiner has previously explained (Office Action of 2 March 2006, page 8), "A CPU is not generally referred to or employed as a storage device."

Applicants' previous argument was not directed to the invention as specified in the claims, and here Applicants have misconstrued the Examiner's response to that argument.

To directly address the present argument, the Examiner respectfully submits that a person of ordinary skill in the art would recognize the processors taught by Shimura (column 4, line 52 – column 5, line 26) as either possessing a code cache or would recognize a code cache as well known in the art. A person of ordinary skill in the art would recognize the purpose of a code cache as "storing code segments" on the client for execution.

Applicants' arguments have been fully considered but have been found unpersuasive.

Regarding claim 3, Shimura teaches that the first format is other than the native execution format of the CPU of the client (column 5, lines 15-26). A compiler is functionally equivalent to a "transformation engine to transform the client application from the first format to the native binary format of the CPU of the client".

Regarding claim 4, Shimura does not explicitly teach that the first format is a source code text format of a programming language and the transformation manager comprises a compiler that compiles and links the client application into a native binary format of the CPU of the client. However, Shimura does explicitly teach that the first format is a "Java program in the form of the virtual machine computer program prepared as an applet on the web page" (column 4, lines 28-30) which is compiled using a Java™ compiler (column 4, lines 42-51 and throughout). It would have been obvious to a person of ordinary skill in the art that the term "Java applet" commonly refers to source code in a text format intended for use in a web page and that source code in a text format is well known input to a typical compiler. It therefore would have been obvious to a person of ordinary skill in the art to implement Shimura's system where the first format is a

source code text format of a programming language and compiling that source code into a native binary format of the CPU of the client.

Regarding claim 5, Shimura teaches that the transformation manager comprises a just-in-time compiler (column 5, lines 8-15).

Regarding claim 6, Shimura's teaching regarding class files (column 7, line 42 – column 8, line 18) would be obvious to a person of ordinary skill in the art at the time of Applicants' invention as functionally equivalent to "code segments". It would be obvious to a person of ordinary skill in the art at the time of Applicants' invention to implement this functionality with a client code manager that requests needed segments from the server and to branch into the received code segment. Indeed, this is the functionality implied by Shimura (column 7, line 57 – column 8, line 13) although the obvious details of implementation are omitted.

Regarding claim 7, Shimura does not explicitly recite the steps of adjusting branch instructions to link into and out of received code segments as recited. Shimura implies this functionality (column 7, line 42 – column 8, line 18; column 9, line 63 – column 10, line 9). Official notice is taken that the need to link code that is compiled in sections is well known. It would have been obvious to a person of ordinary skill in the art at the time of Applicants' invention, in combination with his own knowledge of the particular art, to adequately support linking sections of compiled code by adjusting the branch instructions. Failure to do so would create an inoperable system, as would be recognized as well known by a person of ordinary skill

in the art. Applicants have not traversed this use of Official Notice and as such, it is regarded as admitted prior art. See MPEP 2144.03 (C).

Claim 8 recites what is generally known in the art as "garbage collection". Official notice is taken that Java™ and the Java™ virtual machine support garbage collection. It would have been obvious to a person of ordinary skill in the art at the time of Applicants' invention, in combination with his own knowledge of the particular art, to implement the system taught by Shimura using garbage collection because of the well-known advantages of garbage collection, such as ease of programming and recovery unused memory.

Claims 9-13 recite the server portion of the system of claims 1-5 and are rejected for the same reasons given above for claims 1-5.

Claims 14-17 recite the client portion of the system of claims 1 and 6-8 and are rejected for the same reasons given above for claims 1 and 6-8.

Claims 18-22 recite the methods performed by the system of claims 1-7 and are rejected for the same reasons given above for claims 1-7.

Claims 23-24 recite a computer program product and system according to claims 1-7 and are rejected for the same reasons given above for claim 1.

In response to the rejections of independent claims 9, 14, 18, and 23, Applicants reiterate the arguments shown above pertaining to claim 2. Those arguments are traverses as shown above in regard to claim 2.

Conclusion

2. **THIS ACTION IS MADE FINAL.** Applicant is reminded of the extension of time policy as set forth in 37 CFR 1.136(a).

A shortened statutory period for reply to this final action is set to expire THREE MONTHS from the mailing date of this action. In the event a first reply is filed within TWO MONTHS of the mailing date of this final action and the advisory action is not mailed until after the end of the THREE-MONTH shortened statutory period, then the shortened statutory period will expire on the date the advisory action is mailed, and any extension fee pursuant to 37 CFR 1.136(a) will be calculated from the mailing date of the advisory action. In no event, however, will the statutory period for reply expire later than SIX MONTHS from the mailing date of this final action.

Any inquiry concerning this communication or earlier communications from the examiner should be directed to Jason Proctor whose telephone number is (571) 272-3713. The examiner can normally be reached on 8:30 am-4:30 pm M-F.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Paul Rodriguez can be reached at (571) 272-3753. The fax phone number for the organization where this application or proceeding is assigned is (571) 273-8300.

Any inquiry of a general nature or relating to the status of this application should be directed to the TC 2100 Group receptionist: 571-272-2100. Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR)

system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free).

Jason Proctor
Examiner
Art Unit 2123

jsp

PAUL RODRIGUEZ
SUPERVISORY PATENT EXAMINER
TECHNOLOGY CENTER 2100

8/23/06